## Primer Ejercicio Loops


## Descripción

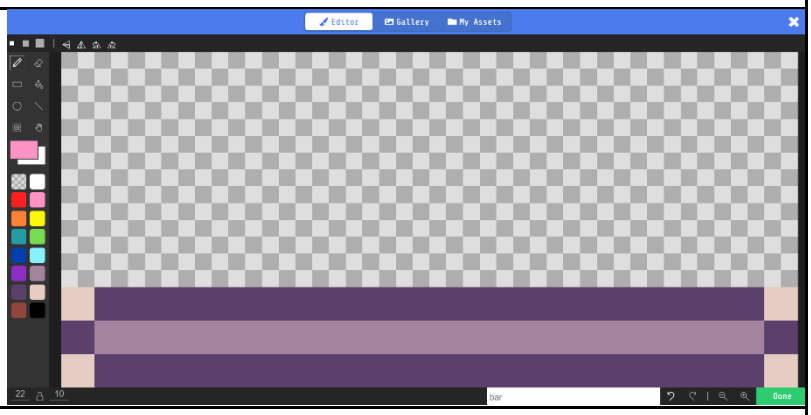
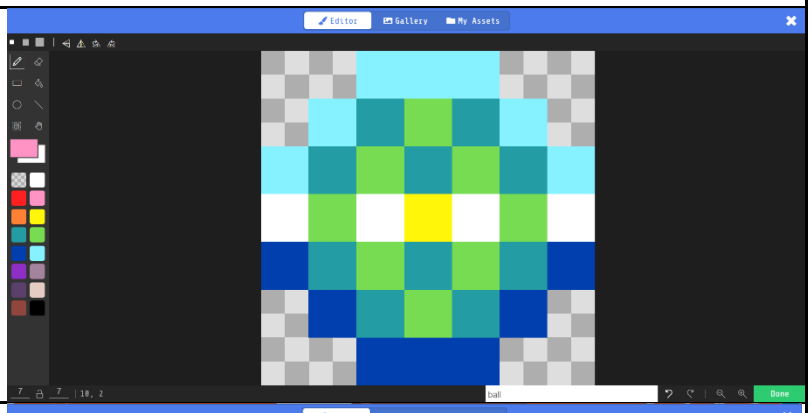
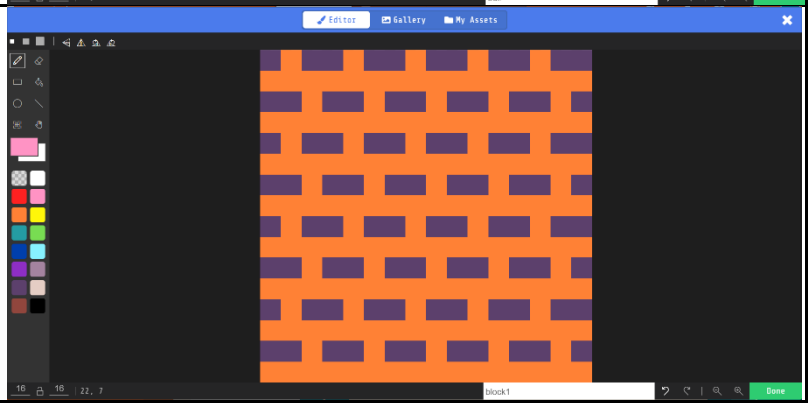
En este proyecto realizaremos un juego muy clásico que es “Block Out”.
El objetivo principal del videojuego es destruir bloques con una pelota evitando que caiga con una barra horizontal.

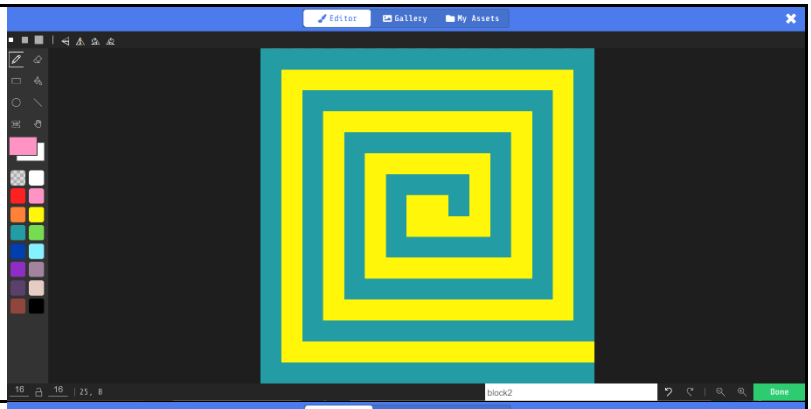
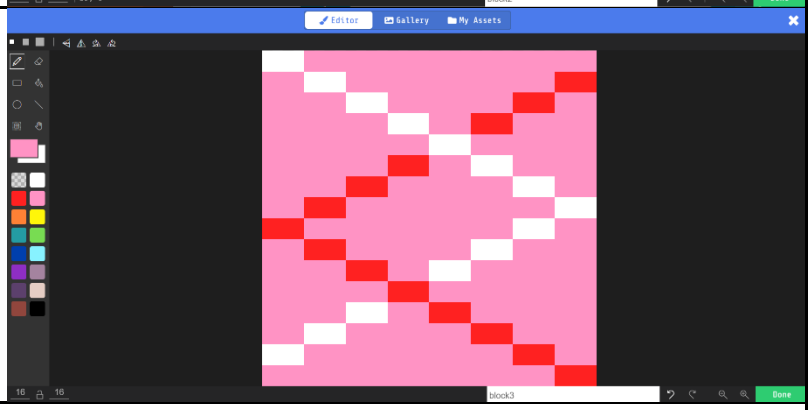
Para ello accederemos a **MakeCode Arcade** y realizaremos las operaciones necesarias.

## Objetivos de programación y diseño

- Crear un Sprite de la barra del jugador.
- Crear un Sprite de la pelota.
- Crear distintos Sprites de bloques.
- Crear Mecánica de rebote de la pelota en los bloques y hacer ganar un punto.
- Crear Mecánica de rebote de la pelota en la barra.
- Crear Mecánica de creación y de colocación de bloques en el escenario de forma automática.

## Programación del juego

| |
|---|
| **Aquí os dejamos el enlace con parte de la programación y Assets hechos.**<br>**https://makecode.com/_Vgd64v9qAHAJ** |
| **CREACIÓN DE ASSETS** |
| **CREACIÓN SPRITE PRINCIPAL** |
| Te recomendamos utilizar una matriz de 22x10 px para el Sprite de bar.<br>  |
| **CREACIÓN SPRITE ADICIONALES** |
| Te recomendamos utilizar una matriz de 7x7 px para el Sprite de ball.<br>  |
| Te recomendamos utilizar una matriz de 16x16 px para el Sprite de block1.<br>  |

| | |
|---|---|
| Te recomendamos utilizar una matriz de 16x16 px para el Sprite de block2. |  |
| Te recomendamos utilizar una matriz de 16x16 px para el Sprite de block3. |  |

## PROGRAMACIÓN PRINCIPAL

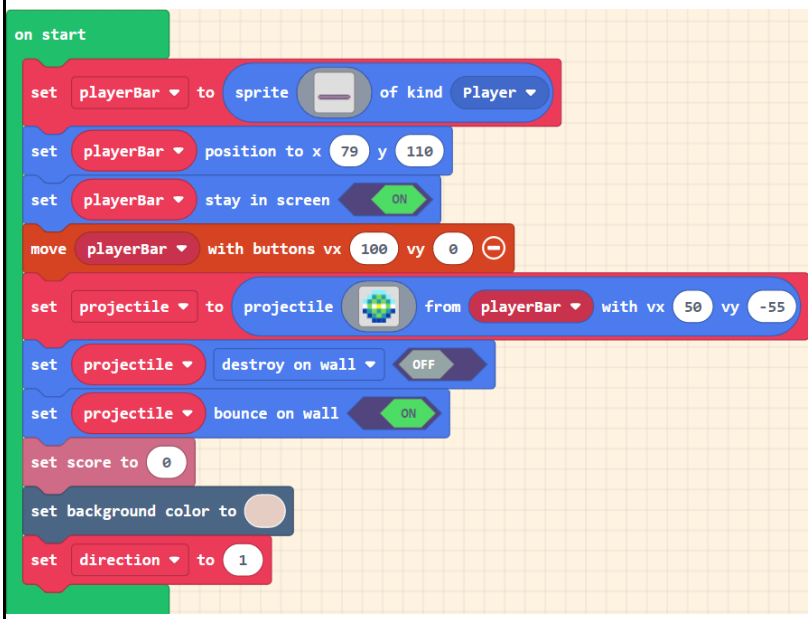
### CREACIÓN INICIO DEL JUEGO

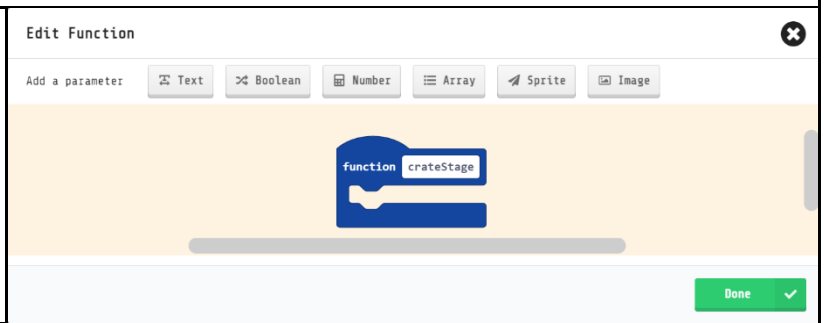
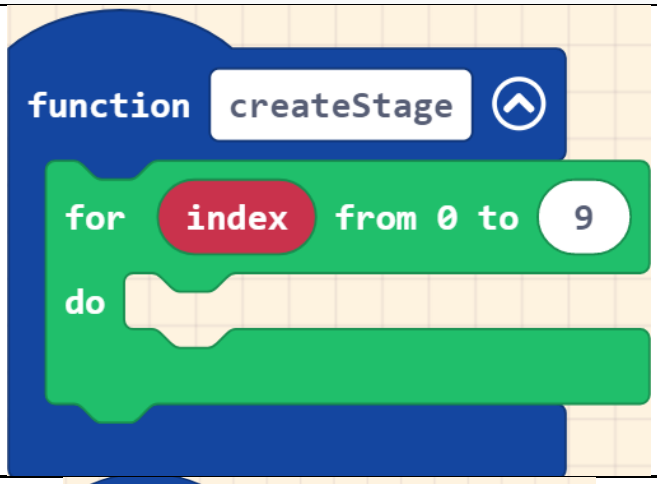
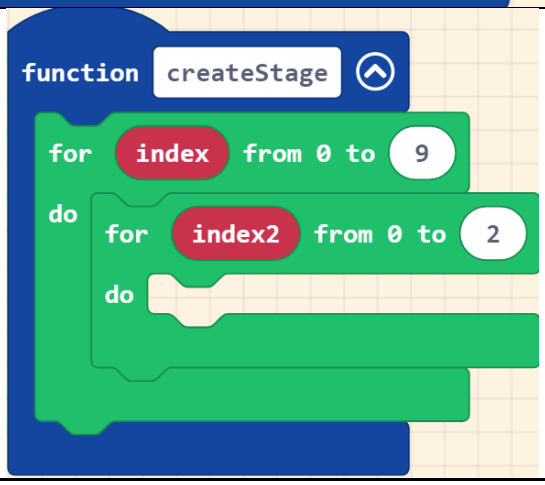
Para comenzar, crearemos un bloque on start y colocaremos en su interior:

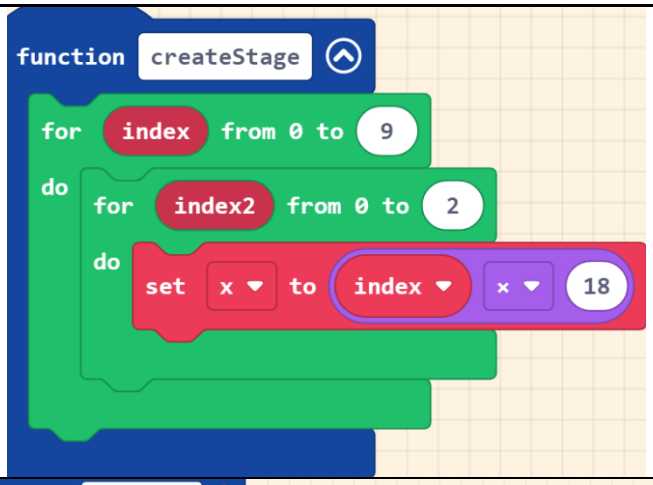
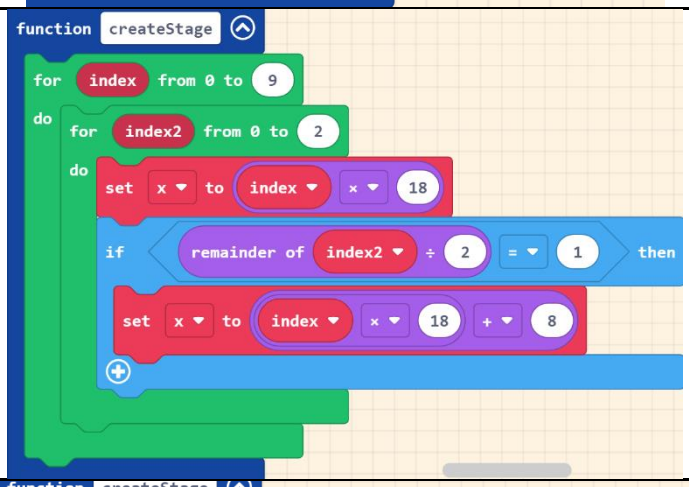
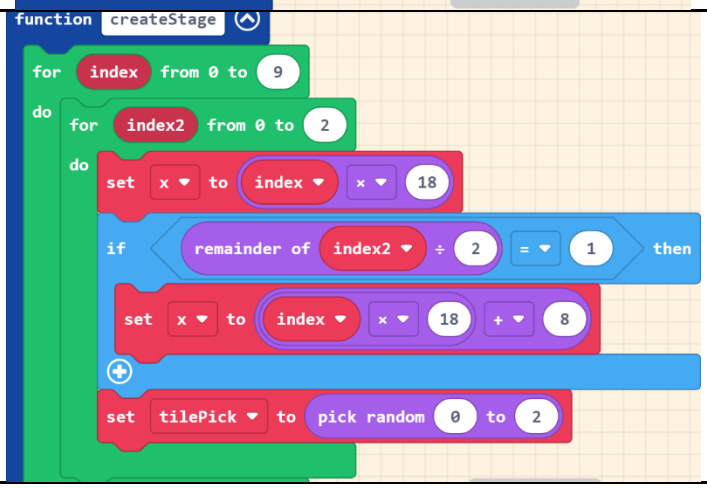
Crearemos playerBar y le daremos nuestro Sprite bar y le asignaremos la condición Player. Lo colocaremos en "x" 79 e "y" 110. También le diremos que no salga de la pantalla y que tenga una velocidad horizontal de 100 y de vertical 0 para que solo se mueva en el eje "x".

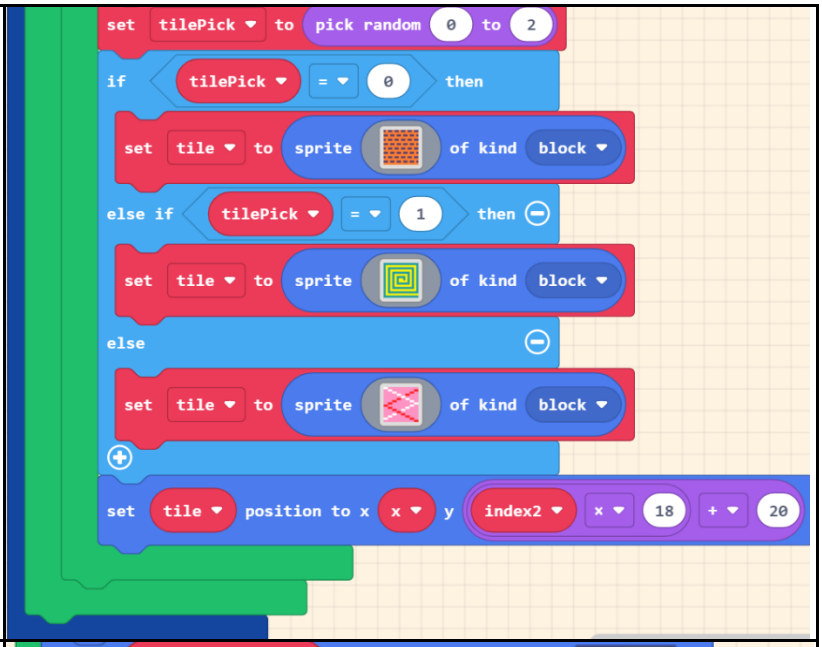
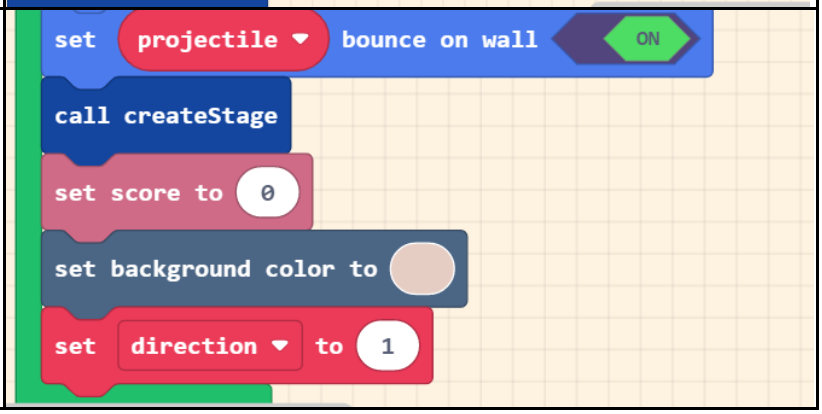
Haremos lo mismo con la pelota pero cambiando su velocidad, su condición, su posición, rebote y que no se destruya.

Set score to lo establecemos en 0, set background color to le ponemos el color que queramos.



| | |
|---|---|
| Por último, creamos la variable direction y le damos el valor 1. | |
| **CREACIÓN FUNCIÓN createStage** | |
| Vamos a crear nuestro escenario, pero en vez de colocar los bloques uno a uno, le vamos a decir a juego que los crea y que después los coloque uno al lado de otro. Lo metemos en una función createStage. |  |
| Vamos a meter un bucle for y el index será de 0 a 9. Con esto vamos a crear el número de bloques que va a contener las filas. Un total de 10. En informática no se comienza a contar desde el 1 como hacemos normalmente, por eso el bucle, comienza desde el 0 y termina en 9, ahí están los 10 números. |  |
| Ahora, para crear el número de filas que va a tener nuestro nivel, le vamos a meter a nuestro bucle for otro y el index2 le decimos que estará entre 0 y 2. Así le decimos al juego que habrá 3 filas. |  |

| | |
|---|---|
| Primero le vamos a decir que, a cada index, lo multiplique por 18 (16 de la medida de nuestro bloque y 2 más para que tenga separaciones entre bloques) para establecer el espacio de nuestro bloque. |  |
| Acto seguido, le vamos a colocar un condicional if y le vamos a decir que si el resto de la división entre index2 y 2 es igual a 1, le decimos que a x le multipliquemos index x 18 y le sumemos 8. Con esto logramos que pase a la siguiente fila y tenga su espacio en vertical. |  |
| Teniendo la base de nuestra creación de filas y bloques por fila vamos a decirle que elija los bloques de forma aleatoria.<br>Para ello empezaremos creando la variable tilePick y le pondremos un valor aleatorio entre 0 y 2. |  |

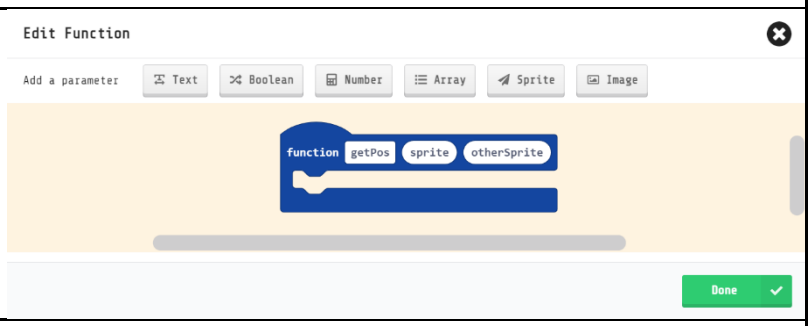
| | |
|---|---|
| Justo después, le decimos que, según el número aleatorio obtenido en la variable, que coloque elija un bloque u otro, que le asigne block y que lo coloque en la posición correspondiente en x, que será en función al número index2 que tengamos en ese momento. |  |
| Por último, una vez terminada la función, la incluiremos en el inicio justo después de programar la bola y antes de declarar el score. |  |

## CREACIÓN MECÁNICA PELOTA REBOTA CON LA BARRA

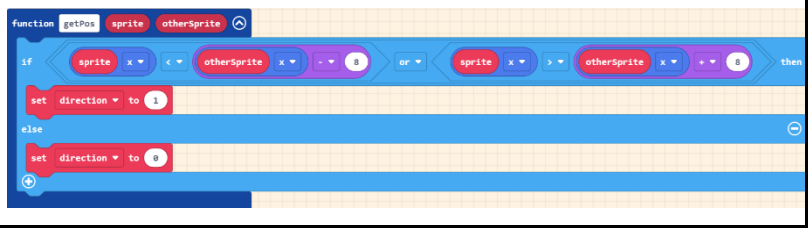
Esta mecánica es sencilla, le vamos a decir nuestro juego que en el momento que nuestra pelota (Projectile) toque la barra (Player) que mantenga su velocidad en x pero que cambie su velocidad en y justo en el sentido contrario. Para conseguirlo, solo hay que multiplicar su velocidad en y por -1.


## CREACIÓN MECÁNICA PELOTA REBOTA CON UN BLOQUE

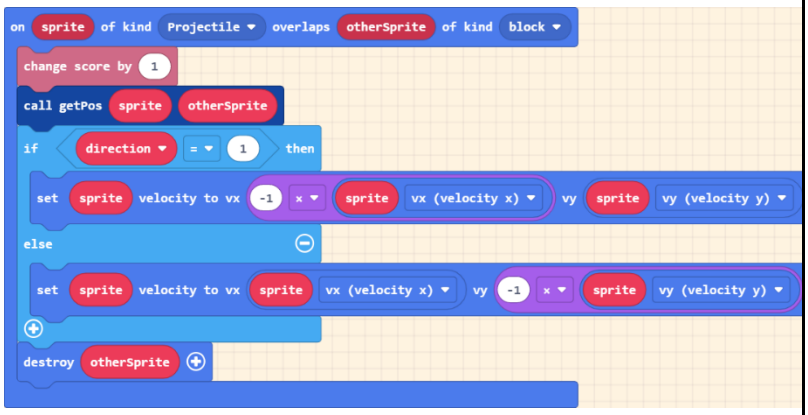
Para empezar, vamos a crear una función que nos diga donde a golpeado la pelota. Para esto, en la función incluiremos 2 parámetros de tipo Sprite y le ponemos de nombre sprite (pelota)y otherSprite (bloque).

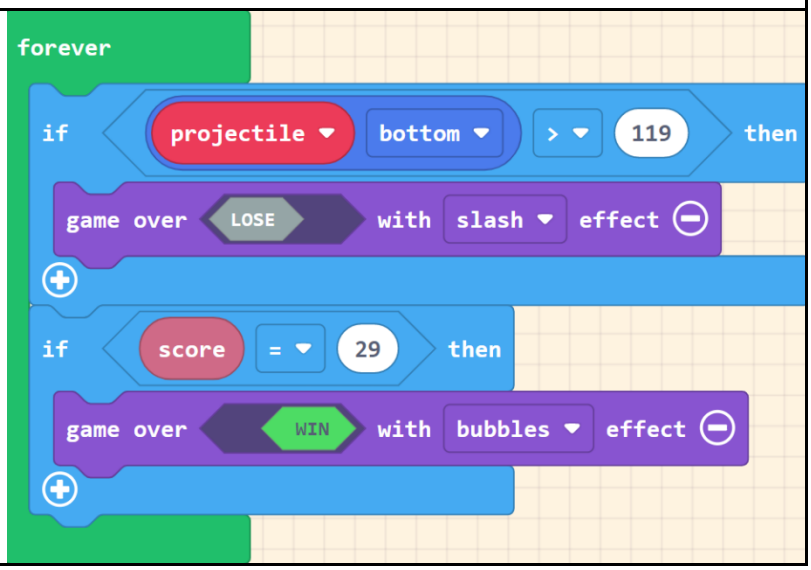



Dentro haremos las siguientes comprobaciones.
Si la pelota colisiona contra la esquina izquierda del bloque o esquina derecha, establecemos direction a 1, si no la establecemos a 0.




Una vez realizada la función vamos a decirle al juego que cuando la pelota colisione con el bloque haga lo siguiente:
Primero que le sume un punto al jugador.
Luego que realice la comprobación de antes mencionada en la función.
Por último, si direction es igual a 1, le decimos que multiplique la vx por -1 para que cambie de dirección en horizontal, pero que prosiga con la misma vy y si no, multiplicamos por -1 la vy para que cambie la dirección en vertical y siga con la misma vx.



| CREACIÓN MECÁNICA DE FINAL DEL JUEGO | |
|---|---|
| Para esta mecánica, comprobaremos cuando en todo momento lo siguiente:<br><br>Primero, si la pelota desciende más de 119 el juego se acaba perdiendo.<br><br>Segundo, si la puntuación es igual a 29, el juego se acaba y ganamos. |  |

Gracias a esta programación hemos hecho un “Block Out” muy básico. Hemos aprendido a realizar un escenario automáticamente a través de los bucles for, también hemos aprendido a realizar comprobaciones de colisiones y aplicarles distintos efectos. Ahora, es tu turno de personalizarlo y añadirle contenido. Aquí te dejamos el nuestro para que te inspires un poco: **https://makecode.com/_Pj54xtFvfddj**

## Glosario

**If-Else**: Sentencia condicional que si se cumple ejecuta una secuencia de instrucciones y otra que omite.

**Operadores de comparación**: Son operadores que comparan un valor a otro y se introducen dentro de una condición.

**Variables**: Es un espacio asociado a un identificador, en ese espacio hay un valor que puede ser modificado.

**Funciones**: Es un subprograma que recoge un conjunto de instrucciones y pueden ejecutarse desde el programa principal haciendo una llamada a él.

**Aceleración**: Es la variación de velocidad por unidad de tiempo.

**Velocidad**: Es una magnitud física que relaciona la posición con el incremento de tiempo.

**Muros**: Objetos o espacios donde los distintos elementos del juego no pueden atravesar.

**Puntuación**: Puntos totales que obtiene un jugador al realizar ciertas interacciones.